UNIVERSITY OF CAMBRIDGE
Department of Engineering

Reinforcement Learning for Clinical Decision Support

Author Name: Aniruddh Raghu
Supervisor: Dr Sumeetpal Singh

Date: Sunday 3rd June, 2018

I hereby declare that, except where specifically indicated, the work submitted herein is my own original work.

Signed: _____ Date: _____

Technical Abstract

Sepsis (severe infection with organ failure) is a dangerous condition that is a leading cause of patient mortality, and is clinically challenging to treat. In the medical literature, there is no well-established treatment strategy for sepsis, and clinicians lack decision support tools to assist them when treating patients with sepsis.

This work explores the use of reinforcement learning (RL) to provide medical decision support for sepsis treatment. It considers the use of RL algorithms and observational data from intensive care units (ICUs) to deduce improved medical treatment strategies (or *policies*) for sepsis. Two areas are addressed in this work. Firstly, an empirical procedure to evaluate sepsis treatment strategies is developed. Secondly, the use of continuous state-space model-based RL to discover improved medical treatment policies for sepsis is investigated. The evaluation procedure considered in this work is used to assess the treatment policies learned with model-based RL.

The empirical evaluation methodology developed uses Off-Policy Evaluation (OPE) to assess medical treatment strategies. OPE requires modelling the decision making strategy used by clinicians and key statistics of the underlying environment from which patient physiological data are generated. Reliable evaluation is obtained using a k-Nearest Neighbours model to represent clinician behaviour and the Fitted Q Iteration algorithm to represent environmental statistics. This approach could be used more generally for OPE in other domains.

The model-based RL procedure in this work firstly constructs a physiological model for patients with sepsis (environment modelling), and then uses this model to identify suitable treatment strategies (policy search). Both tasks require special considerations given the highly stochastic nature of medical data, and the need for the discovered treatment strategies to remain interpretable. The resulting solution proposes an algorithm, using neural networks, to learn a suitable treatment strategy. This treatment strategy appears to improve on what clinicians follow, and insights from it could be used to improve the standard of patient care.

Table of contents

Nomenclature

Roman Symbols

a_t	Action at time t
$Q^\pi(s, a)$	Action value function for policy π from state s and action a
r_t	Reward at time t
s_t	State at time t
V^π	Value of policy π
$V^\pi(s)$	Value function for policy π from state s

Greek Symbols

γ	Discount factor
π	Policy
π_b	Behaviour policy
π_e	Evaluation policy

Other Symbols

$\mathcal{A}$	Action-space
$\mathcal{D}$	Dataset
$\mathcal{S}$	State-space

Acronyms / Abbreviations

AM	Approximate Model
BNN	Bayesian Neural Network
FQI	Fitted Q Iteration

ICU	Intensive Care Unit
IS	Importance Sampling
KBRL	Kernel-Based Reinforcement Learning
kNN	k-Nearest Neighbours
LR	Logistic Regression
MDP	Markov Decision Process
MSE	Mean Squared Error
NN	Neural Network
OPE	Off-Policy Evaluation
PG	REINFORCE Policy Gradient algorithm
PHWDR	Per-Horizon Weighted Doubly Robust
PHWIS	Per-Horizon Weighted Importance Sampling
PPO	Proximal Policy Optimization algorithm
ReLU	Rectified Linear Unit activation
RF	Random Forest
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SOFA	Sequential Organ Failure Assessment score
WDR	Weighted Doubly Robust
WIS	Weighted Importance Sampling

Chapter 1

Introduction

Sepsis (severe infection with organ failure) is a dangerous condition that is a leading cause of patient mortality [1] and costs hospitals billions of pounds to treat [2]. Sepsis is often managed by giving patients intravenous fluids and vasopressors. Different dosage strategies for these two interventions can greatly affect patient outcomes, which demonstrates how important treatment decisions are [3]. However, clinicians still lack decision support tools to assist them when treating patients with sepsis [4].

This work explores the use of reinforcement learning (RL) to provide medical decision support for sepsis treatment. The study considers the use of RL algorithms and observational data from intensive care units (ICUs) to deduce improved medical treatment strategies (or *policies*) for sepsis. The use of RL in this work is well-motivated over supervised learning, primarily because the objective is to discover improved strategies from suboptimal training examples, and RL algorithms are designed for this particular scenario.

This study develops on the ideas explored in Raghu et al. [5], and examines two related directions. The objectives of this work are to:

1. Develop an empirical evaluation procedure to assess the quality of medical treatment strategies for sepsis.

2. Investigate whether continuous state-space model-based RL algorithms can be used to find improved treatment policies for sepsis.

Developing a reliable evaluation procedure is an important task, because without such a methodology, it is difficult to assess whether a learned treatment strategy improves on what is currently followed by doctors. However, evaluating medical treatment strategies is challenging. Common approaches to evaluation in RL include deploying a decision making strategy in the real-world or in a simulator and observing its performance. Real-world deployment is not appropriate for problems in healthcare, because a medical treatment strategy cannot be used in a hospital prior to knowing whether it is safe, for

ethical reasons. Developing a reliable simulator for evaluation is also challenging, as it requires a highly accurate physiological model, which is difficult to construct. It is therefore necessary to use a method of evaluation that relies on the observational data available already – this is the problem of Off-Policy Evaluation (OPE) [6].

There has been significant prior research in OPE [6–9], but past empirical work has focused only on synthetic benchmark scenarios. In real-world observational studies (such as the one considered here), OPE methodologies face two key problems: unknown *behaviour policies* and hard-to-estimate *approximate model terms*. Both the behaviour policy and approximate model terms are required for accurate OPE. In medical domains, the behaviour policy is the probability of a doctor deciding on the chosen medical intervention, given the patient's physiological state. When this is unknown, it can be estimated from data, but estimation errors lead to flawed OPE. Approximate model terms represent key statistics of the underlying environment from which the data are generated. They can be hard to estimate in complex state-spaces (as examined in this work), and as with behaviour policies, estimation errors result in inaccurate OPE. The first objective of this work is therefore to develop a methodology to accurately estimate behaviour policies and approximate model terms for the sepsis domain and use these estimates in OPE to evaluate medical treatment strategies.

The second objective of this study — exploring the use of model-based RL with continuous state-spaces — is a well-motivated direction of work. Prior research has examined model-free RL with continuous state-spaces [5] and model-based RL with discrete state-spaces [10] for discovering sepsis treatment policies. However, both model-free RL and discretised state-space modelling have important limitations. Model-free RL can be data-inefficient and typically has poor sample complexity [11]; model-based RL can improve on these shortcomings. Discrete state-space models cause information to be lost from the original features (due to the discretisation process) and this can worsen the quality of the discovered treatment policies. Continuous state-space models are more challenging to train, but they utilise all available information about patient physiology, which could lead to finding improved treatment strategies.

This dissertation addresses these two objectives as follows. Chapter 2 provides background information about Markov Decision Processes (MDPs), Reinforcement Learning and Off-Policy Evaluation. It introduces the specific MDP framing used for the sepsis domain. Chapter 3 addresses OPE, including behaviour policy estimation and developing approximate models. Chapter 4 considers model-based RL, and explores building physiological models for use in model-based RL algorithms and developing a procedure for *policy search* – using the physiological model to find treatment policies. This chapter provides a qualitative and quantitative assessment of the discovered policies. The quantitative assessment uses the OPE procedure developed in Chapter 3.

Chapter 2

Background

This chapter provides background information about the reinforcement learning problem, Markov Decision Processes and Off-Policy Evaluation. Information is given about the specific modelling for the sepsis treatment problem.

2.1 Reinforcement Learning and Markov Decision Processes

In the reinforcement learning (RL) problem, an agent's interaction with an environment can be represented by a Markov Decision Process (MDP). An MDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, R, P, P_0, \gamma \rangle$, where $\mathcal{S}$ is the state-space, $\mathcal{A}$ is the action-space, $R(s, a, s')$ is the reward function, governing the reward r received when taking action a in state s and transitioning to state s', $P(\cdot|s, a)$ is the transition probability distribution, P_0 is the initial state distribution, and $\gamma \in [0, 1)$ is the discount factor. Define the *policy*, π, as a mapping from states to actions: $\pi(a|s)$ represents the probability of taking action a in state s.

Let $H := (s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1}, s_T)$ be a trajectory generated when following policy π. The T-step return of a trajectory H, $R_{0:T-1}(H)$, is defined as follows:

$$R_{0:T-1}(H) = \sum_{t=0}^{T-1} \gamma^t r_t$$

Define the value of a policy π, V^π, as the expected return over trajectories it generates, with the expectation being taken over the probability distribution of trajectories under policy π:

$$V^\pi = \mathbb{E}_{H \sim P_H^\pi} \Big[R_{0:T-1}(H) \Big]$$

.

Let the value and action value functions of a policy π at a state s or state-action pair (s, a) be $V^\pi(s)$ and $Q^\pi(s, a)$ respectively. These are defined as the expected return of a trajectory starting at state s or state-action pair (s, a), and then following policy π, as follows:

$$V^\pi(s) = \mathbb{E}_{H \sim P_H^\pi}\Big[R_{0:T-1}(H) | s_0 = s\Big]$$
$$Q^\pi(s, a) = \mathbb{E}_{H \sim P_H^\pi}\Big[R_{0:T-1}(H) | s_0 = s, a_0 = a\Big]$$

From these definitions, we can relate $V^\pi(s)$ and $Q^\pi(s, a)$ as follows:

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)}\Big[Q^\pi(s, a)\Big]$$

We can relate V^π and $V^\pi(s)$ by taking an expectation over the starting state s_0:

$$V^\pi = \mathbb{E}_{s_0 \sim P_0}\Big[V^\pi(s_0)\Big]$$

The goal of an RL agent is to learn an optimal policy $\pi^*(s)$, defined as one that maximises its expected discounted future return:

$$\pi^*(s) = \arg\max_\pi V^\pi(s)$$

In *model-based* RL, learning proceeds by firstly modelling the transition distribution P and then using this to find π^*. In *model-free* RL, π^* is learned directly without modelling P. Prior work has considered model-free RL in continuous state-spaces [5]; this work considers the use of model-based RL.

2.2 Off-Policy Evaluation

The Off-Policy Evaluation (OPE) problem considers estimating the value V^{π_e} of an evaluation policy π_e given a set of trajectories $\mathcal{D} = \{H^{(i)}\}_{i=1}^n$ generated independently by following a (typically distinct) behaviour policy π_b. The estimator $\hat{V}^{\pi_e}$ is desired to have low mean squared error (MSE), defined as follows:

$$\text{MSE}(V^{\pi_e}, \hat{V}^{\pi_e}) = \mathbb{E}_{P_H^{\pi_b}}\Big((V^{\pi_e} - \hat{V}^{\pi_e})^2\Big)$$

When there are trajectories available from following π_e, perhaps from using π_e in a simulator, the OPE problem reduces to one of on-policy evaluation, where $\pi_e = \pi_b$. To find $\hat{V}^{\pi_e}$ in this setting, the Monte Carlo estimator can be used:

$$\hat{V}^{\pi_e}_{\text{MC}} = \frac{1}{N}\sum_{i=1}^{N} R_{0:T-1}(H^{(i)})$$

Prior work in OPE has approached the problem of estimating the quantity V^{π_e} using one or both of the techniques of Importance Sampling (IS) and Approximate Model (AM) estimation [8]. These methods are now explained further.

2.2.1 Importance Sampling (IS)

The IS approach to evaluation uses importance sampling to adjust for the difference between the probability of a trajectory H under the behaviour policy π_b and the probability under the evaluation policy π_e. To form the IS OPE estimator, first define the *importance weight* ρ_{T-1} to be the ratio of the probability of the first T steps of trajectory H under π_e to the probability under π_b [1]:

$$\rho_{T-1} = \prod_{t=0}^{T-1} \frac{\pi_e(a_t^H|s_t^H)}{\pi_b(a_t^H|s_t^H)}$$

The IS estimator of V^{π_e} can then be defined as follows [6], with i indexing the trajectories in $\mathcal{D}$:

$$\hat{V}^{\pi_e}_{\text{IS}} = \frac{1}{n}\sum_{i=1}^{n} \rho^{(i)}_{T-1} \sum_{t=0}^{T-1} \gamma^t \; r_t^{(i)}$$

Two commonly used estimators in the IS family, which improve on this simple estimator, are the step-wise IS and step-wise Weighted IS (WIS) estimators, defined as follows:

$$\hat{V}^{\pi_e}_{\text{step-IS}} = \frac{1}{n}\sum_{i=1}^{n}\sum_{t=0}^{T-1} \gamma^t \rho_t^{(i)} \; r_t^{(i)}$$

$$\hat{V}^{\pi_e}_{\text{step-WIS}} = \sum_{i=1}^{n}\sum_{t=0}^{T-1} \gamma^t \frac{\rho_t^{(i)}}{\sum_{i=1}^{n} \rho_t^{(i)}} \; r_t^{(i)}$$

These definitions hold for a dataset of trajectories with the same length; an extension to handle trajectories of different length can be found in Doroudi et al. [12] – this is called the *Per-Horizon* extension (resulting in the PHIS and PHWIS estimators).

[1] Assume henceforth that for all state-action pairs $(s, a) \in \mathcal{S} \times \mathcal{A}$, if $\pi_b(a|s) = 0$ then $\pi_e(a|s) = 0$.

The step-IS estimator is an unbiased estimator of V^{π_e}; that is, $\mathbb{E}\left[\hat{V}^{\pi_e}_{\text{step-IS}}\right] = V^{\pi_e}$. However, it suffers from high variance, due to the product of up to T probability ratios, which can become very small or very large depending on π_b and π_e.

The step-WIS estimator is biased, but consistent; that is, for finite n, $\mathbb{E}\left[\hat{V}^{\pi_e}_{\text{step-WIS}}\right] \neq V^{\pi_e}$ but as $n \to \infty$, $\mathbb{E}\left[\hat{V}^{\pi_e}_{\text{step-WIS}}\right] \to V^{\pi_e}$. This estimator has lower variance than step-IS because the importance weights are bounded between zero and one (via the normalisation). However, its variance can often still be unacceptably high [8]. All IS estimators can have significant bias when the behaviour policy is unknown and estimated incorrectly.

2.2.2 Approximate Models (AMs)

In AM estimation, an approximate model M of the underlying MDP is firstly constructed. This is used to find $\hat{V}^{\pi}_{M}(s)$ and $\hat{Q}^{\pi}_{M}(s,a)$, which are estimates of the state and action value functions for policy π under M respectively. The approximate model can be used to directly find V^{π_e}:

$$\hat{V}^{\pi_e}_{\text{AM}} = \frac{1}{n}\sum_{i=1}^{n}\sum_{a\in\mathcal{A}} \pi_e(a|s_0^{(i)})\hat{Q}^{\pi_e}_{M}(s_0^{(i)},a)$$

Depending on the model class and algorithm used to define M, it can be difficult to prove that these estimators are unbiased/consistent; therefore, it may not be possible to trust their estimates in isolation.

2.2.3 Doubly Robust (DR)

DR methods [7, 8] combine IS and AM techniques in order to obtain unbiased, reduced variance estimators. The Weighted Doubly Robust (WDR) estimator, which has demonstrated effective empirical performance [8], is defined as follows:

$$\hat{V}^{\pi_e}_{\text{WDR}} = \sum_{i=1}^{n}\sum_{t=0}^{T-1}\left(\gamma^t w_t^{(i)} r_t^{(i)} - \gamma^t\left(w_t^{(i)}\,\hat{Q}^{\pi_e}_{M}(s_t^{(i)},a_t^{(i)}) - w_{t-1}^{(i)}\,\hat{V}^{\pi_e}_{M}(s_t^{(i)})\right)\right)$$

where $w_t^{(i)}$ is defined as:

$$w_t^{(i)} = \frac{\rho_t^{(i)}}{\sum_{i=1}^{n}\rho_t^{(i)}}$$

2.2.4 OPE in this work

We consider the problem of obtaining reliable OPE for medical treatment strategies using the WIS, AM, and WDR approaches. We firstly define a step-wise extension for the PHWIS estimator to reduce its variance and an extension on the WDR estimator

to handle variable trajectory lengths. We then explore how to estimate the unknown behaviour policy $\pi_b(\cdot|s)$ and the approximate model terms $\hat{Q}_M^{\pi_e}(s,a)$.

2.3 Case study: RL for sepsis treatment strategies

The goal in this work is to explore the use of RL as a medical decision support tool for sepsis treatment. This section outlines how the medical treatment of sepsis is modelled, the specific patient cohort considered, and the related work using RL to identify sepsis treatment strategies.

2.3.1 Markov Decision Process (MDP) formulation

This work adopts the modelling of sepsis treatment presented in Raghu et al. [5], where the medical treatment process for a sepsis patient was framed as an MDP. Discrete timesteps in the MDP are defined as four-hour blocks.

The state-space $\mathcal{S}$ captures patient physiology, with the state at a given timestep s_t representing a patient's physiological state, represented as a continuous vector of demographic features, vital signs, and lab values. In order to capture temporal patterns in a patient's physiology, the state representation used in this work concatenates the previous three timesteps' raw state information to the current time's state vector, resulting in a vector of length 198. A full list of the features in the state space is provided in Appendix A.

The action-space, $\mathcal{A}$, is of size 25 and is discretised over dosage amounts of vasopressors and IV fluids, two drugs commonly given to sepsis patients. The discretisation includes a specific dosage for 'no drug given'.

The reward r_t is clinically guided, with positive rewards being issued at intermediate timesteps for improvements in a patient's wellbeing (with improvement being defined by reductions in certain severity scores), and negative rewards for deterioration. At the terminal timestep of a patient's trajectory, a reward is assigned that is positive in the case of survival, and negative otherwise.

2.3.2 Patient cohort

Data for the patients in the cohort considered were obtained from the Medical Information Mart for Intensive Care (MIMIC-III v1.4) database [13]. The dataset used was the same as that in Raghu et al. [5]. The cohort of patients considered fulfill the Sepsis-3 criteria [14] – summary information is presented in Table 2.1. For each patient, at every timestep for the duration of their stay in the ICU, we have information about their physiological state (as a continuous vector) and the treatment they received from

clinicians (in terms of IV fluid and vasopressor dose). We also know their eventual outcome (survival/mortality).

Table 2.1 Summary statistics for the patient cohort.

	% Female	Mean Age	Hours in ICU	Total Population
Survivors	43.6	63.4	57.6	15,583
Non-survivors	47.0	69.9	58.8	2,315

2.3.3 Prior work on RL for sepsis treatment

Komorowski et al. [10] and Raghu et al. [5] investigated using RL to discover sepsis treatment strategies, using the same cohort as this work. Komorowski et al. [10] tackled this problem by using discretised state-space, discretised action-space models, and Q-value iteration to discover suitable medical treatment strategies. Raghu et al. [5] extended this work to consider continuous state-space models.

Continuous state-space models are in general preferable to discrete state-space models as they do not lose information present in the original state-space via a discretisation step [5]. However, it is more challenging to apply RL algorithms in the fully continuous setting; tabular methods with proven convergence properties [15] can no longer be used. Raghu et al. [5] specifically investigated the use of model-free deep RL to discover medical treatment strategies in continuous state-space settings, using deep Q-learning [16], which involves the use of neural networks to discover an optimal policy.

In this work, we approach the problem of learning medical treatment strategies for sepsis using model-based RL, which may suffer from model bias, but has better generalisation and data-efficiency than model-free RL [11]. We also examine empirical methods to reliably evaluate the discovered medical treatment strategies, which has not been explored in detail beforehand.

Chapter 3

Off-Policy Evaluation

This chapter considers how to develop a reliable evaluation procedure for sepsis treatment strategies. As described in Chapter 1, we must assess the quality of medical treatment policies using the observational data available already – this is Off-Policy Evaluation (OPE). We consider evaluating policies using three OPE methods – Weighted Importance Sampling (WIS), Approximate Model (AM) and Weighted Doubly Robust (WDR). Using these methods requires an estimate of the behaviour policy in the dataset, π_b, and an estimate of the action value function for the evaluation policy, $\hat{Q}^{\pi_e}(s, a)$.

The section begins by defining OPE estimators that can be used with a dataset of differing trajectory lengths; this extends work by Doroudi et al. [12]. We then address the tasks of behaviour policy and approximate model estimation, and conclude with assessing the quality of the developed OPE procedure.

3.1 Per-Horizon OPE estimators

OPE estimators such as the Weighted Importance Sampling (WIS) estimator, defined in Section 2.2, are designed to work with a dataset of trajectories of the same length, or horizon. As the lengths of trajectories vary in the dataset considered, these estimators must be extended to handle trajectories with different horizons.

Doroudi et al. [12] defined the Per-Horizon Weighted Importance Sampling (PHWIS) estimator as follows:

$$\hat{V}^{\pi_e}_{\text{PHWIS}} = \sum_{l \in \mathcal{L}} W_l \, \frac{1}{\sum_{\{\tau_i | T_i = l\}} \rho^{(i)}_{T_i - 1}} \sum_{\{\tau_i | T_i = l\}} \rho^{(i)}_{T_i - 1} \sum_{t=0}^{T_i - 1} \gamma^t \, r^{(i)}_t$$

where $\mathcal{L}$ is the set of all trajectory lengths, and W_l is the fraction of the total number of trajectories n with length equal to l:

$$W_l = \frac{|\{\tau_i | T_i = l\}|}{n}$$

This estimator has high variance; we define a lower variance equivalent by considering a step-wise version:

$$\hat{V}^{\pi_e}_{\text{step-PHWIS}} = \sum_{l \in \mathcal{L}} W_l \sum_{\{\tau_i | T_i = l\}} \sum_{t=0}^{T_i - 1} \frac{\rho_t^{(i)}}{\sum_{\{\tau_i | T_i = l\}} \rho_t^{(i)}} \gamma^t \; r_t^{(i)}$$

We now introduce approximate model terms into the estimator (to reduce its variance) and form the Per-Horizon Weighted Doubly Robust (PHWDR) estimator. Define $\hat{V}^{\pi_e}_{\text{WDR},\,l}$ as the WDR estimator given all trajectories of length l. We can write this as follows:

$$\hat{V}^{\pi_e}_{\text{WDR},\,l} = \sum_{\{\tau_i | T_i = l\}} \sum_{t=0}^{T-1} \left(\gamma^t w_{t,l}^{(i)} r_t^{(i)} - \gamma^t \left(w_{t,l}^{(i)} \hat{Q}_M^{\pi_e}(s_t^{(i)}, a_t^{(i)}) - w_{t-1,l}^{(i)} \hat{V}_M^{\pi_e}(s_t^{(i)}) \right) \right)$$

with $w_{t,l}^{(i)}$ defined as:

$$w_{t,l}^{(i)} = \frac{\rho_t^{(i)}}{\sum_{\{\tau_i | T_i = l\}} \rho_t^{(i)}}$$

Then, with W_l as defined before:

$$\hat{V}^{\pi_e}_{\text{PHWDR}} = \sum_{l \in \mathcal{L}} W_l \; \hat{V}^{\pi_e}_{\text{WDR},\,l}$$

The step-PHWIS and PHWDR estimators are used for OPE in this work. For clarity, the step-PHWIS estimator is referred to as the PHWIS estimator from this point onwards.

3.2 Behaviour policy estimation

We now consider how to estimate the behaviour policy in the dataset for use in importance sampling (IS). As IS requires the probability of taking an action under the behaviour policy, $\pi_b(a|s)$, our behaviour policy model must represent the full probability distribution of actions, given states. We therefore must decide a model class to represent the behaviour policy that provides *well-calibrated* probability estimates, not just high accuracy; that is, the probability estimates produced by the model should represent true probabilities.

We motivate the discussion by considering the effect of miscalibration on OPE estimates, to illustrate the importance of calibration in OPE. We then consider how to obtain well-calibrated behaviour policy estimates for the specific dataset used in this work.

3.2.1 Importance of model calibration

To analyse the empirical effect of poorly calibrated behaviour policy models, we considered OPE in a synthetic domain, illustrated in Figure 3.1. Our domain was a continuous 2D map ($s \in \mathbb{R}^2$) with a discrete action-space, $\mathcal{A} = \{1, 2, 3, 4, 5\}$, with actions representing a movement of one unit in one of the four coordinate directions or staying in the current position. Gaussian noise of zero mean and specifiable variance was added onto the state of the agent after each action, to provide environmental stochasticity. An agent started in the top left corner of the domain and received a positive reward within a given radius of the top right corner, and a negative reward within a given radius of the bottom left corner. The horizon was set to be 15 in all experiments, to reflect the typical trajectory length in the original medical domain.

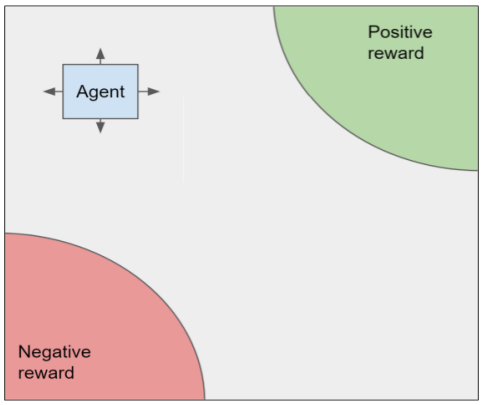


Fig. 3.1 The synthetic domain used to examine the importance of OPE calibration.

To estimate the behaviour policy, we used a k-Nearest Neighbours (kNN) model; we built a kNN data structure on a set of training data points, and at test time, used the points closest in Euclidean distance to compute an empirical probability distribution over the possible actions. The accuracy of the model was varied by changing the number of trajectories and neighbours used for behaviour policy estimation.

Figure 3.2 shows how the average absolute error in the behaviour policy estimation, $\frac{1}{n}\sum_{i=1}^{n} |\pi(a^{(i)}|s^{(i)}) - \hat{\pi}(a^{(i)}|s^{(i)})|$, relates to the fractional error in OPE, for two behaviour policies of differing complexity. The mean and standard deviation of OPE error are shown. Note that the OPE error considers only the error incurred by using an estimated behaviour policy:

$$\text{OPE error} = \frac{V_{\text{WIS},\,\hat{\mu}} - V_{\text{WIS}}}{V_{\text{WIS}}}$$

where $V_{\text{WIS},\,\hat{\mu}}$ refers to the WIS estimate when using an estimated behaviour policy $\hat{\mu}$, and V_{WIS} is the WIS estimate using the true behaviour policy.

This plot shows that the quality of the OPE result is strongly dependent on the quality of the behaviour policy estimation. Average absolute errors in behaviour policy

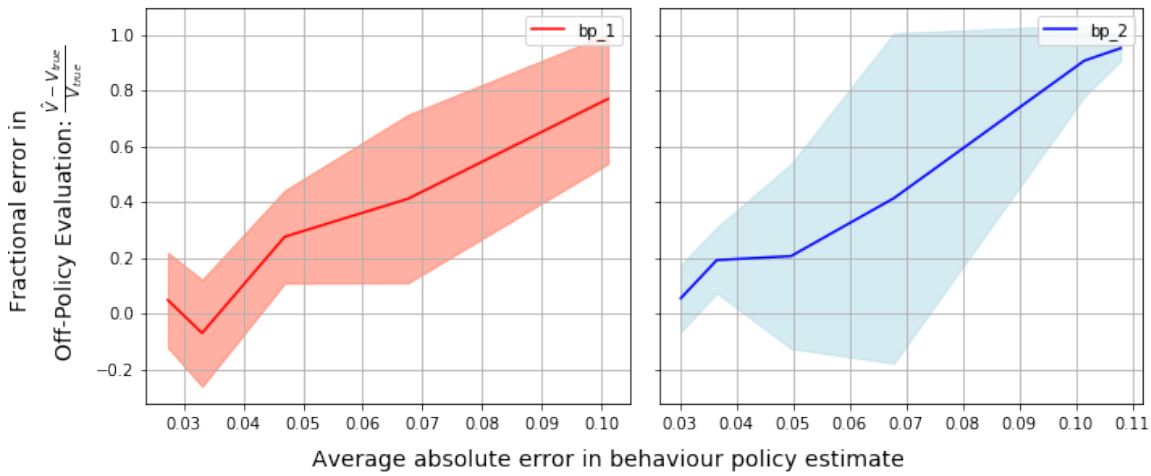


Fig. 3.2 Mean and standard deviation of the fractional error in OPE as a function of the average absolute error in behaviour policy estimation, $\frac{1}{n}\sum_{i=1}^{n}|\pi(a^{(i)}|s^{(i)}) - \hat{\pi}(a^{(i)}|s^{(i)})|$, for two different behaviour policies. Poorly calibrated behaviour policy models lead to highly inaccurate OPE.

models of as small as 0.06 can incur errors of up over 50% in the estimated value. The error incurred worsens as the trajectory length is increased due to the product of (incorrect) terms when forming the importance weight. It is clear that having a well-calibrated model of the behaviour policy is critical for good OPE.

3.2.2 Model calibration in the medical domain

We now consider how to obtain well-calibrated behaviour policy models in the sepsis domain. The calibration of a behaviour policy model can be assessed by computing the distance between the estimated distribution over actions using the model, $\hat{\pi}_b(\cdot|s)$, and the ground-truth distribution over actions, $\pi_b(\cdot|s)$. As $\pi_b(\cdot|s)$ is unknown for this dataset, we proposed to approximate it.

We assumed that $\pi_b(\cdot|s)$ could be determined by considering the empirical distribution over actions formed from the set of neighbours of s, where the neighbours were determined by assessing physiological similarity with s. This makes intuitive sense, as clinicians would be expected to take similar actions for patients with similar physiological states.

We defined similarity of patient states using a 'physiological distance kernel', which was based on Euclidean distance and upweighted informative features of the patient's state. Informative features were the patient's SOFA score, lactate levels, fluid output, mean and diastolic blood pressure, PaO_2/FiO_2 ratio, chloride levels, weight, and age. These are clinically interpretable: the SOFA score and lactate levels provide indications of sepsis severity; careful monitoring of a patient's fluid levels is essential when managing sepsis [17]; and blood pressure indicates whether a patient is in septic shock. These

features were upweighted by a factor of 2 in our distance kernel (where $D = 198$, the dimensionality of our state representation):

$$k(\mathbf{s}, \mathbf{s}') = \sum_{i=1}^{D} w_i (s_i - s_i')^2 \begin{cases} w_i = 2, \text{ for } \textit{informative } i \\ w_i = 1, \text{ otherwise} \end{cases}$$

To find the ground-truth distribution for a given test state, we built a k-Nearest Neighbours (kNN) model with this distance kernel on the test set, and formed an empirical distribution over actions using these k neighbours (considering what actions were taken in those neighbouring k states and then aggregating counts and normalising). We considered 150 neighbours to provide reasonable coverage in the estimate. A Ball Tree data structure, implemented in the scikit-learn library [18], was used to find the kNN.

Behaviour policy models: Several different models were considered, including logistic regression (LR), random forest (RF) with 100 trees, neural network (NN), and an approximate kNN model using random projections [19] (used instead of full kNN as approximate kNN is more computationally efficient). LR and RF were implemented using the scikit-learn library [18]. The approximate kNN model was constructed using the ANNOY library [20]. The neural network was implemented using TensorFlow [21] and had the following architecture:

- Input: patient state for current timestep s_t concatenated with patient state and clinician action for the previous three timesteps. Let this be h_t.
- Fully-connected layer, 64 hidden units, Rectified Linear Unit (ReLU) activation
- Fully-connected layer, 64 hidden units, ReLU activation
- Output layer: fully-connected, 25 output units (probability of each action in the action-space), softmax activation. Let this output be $\hat{\pi}_b$.

This network was trained with stochastic gradient descent and the Adam [22] optimiser. More information about neural networks is provided in Section 4.1

All parametric behaviour policy models (LR, RF, and NN) were trained to predict the clinician action given h_t. We assess the quality of the models using the total variation distance, defined as:

$$\delta\Big(\pi_b(\cdot|s), \hat{\pi}_b(\cdot|s)\Big) = \frac{1}{2} \sum_{a \in \mathcal{A}} |\pi_b(a|s) - \hat{\pi}_b(a|s)|$$

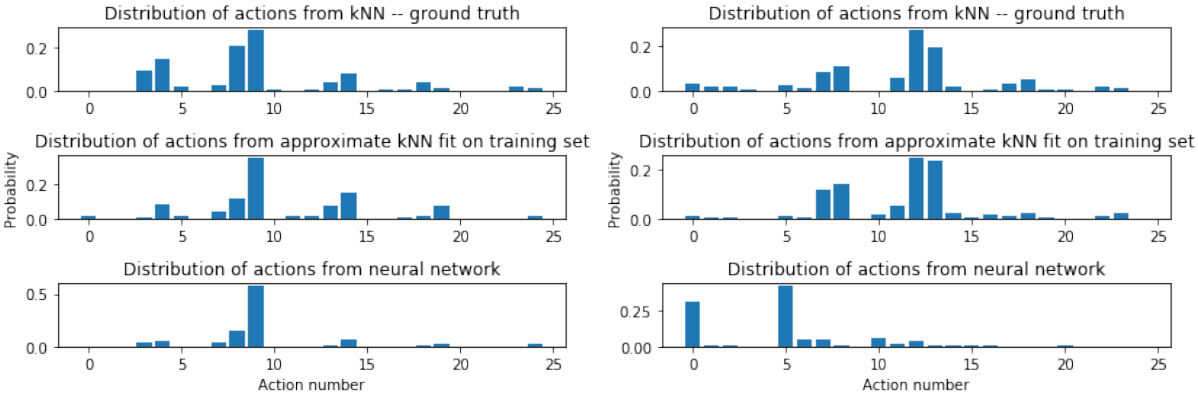
Calibration results: Table 3.1 shows the average total variation distance between the estimated and target behaviour distributions for the different behaviour policy models. Querying the Ball Tree data structure was computationally expensive ($\sim$1 second per query), so the results shown average over 500 sampled test states for patients

at different severities (defined by different SOFA scores). The parametric models are poorly calibrated, especially for sampled states with high severity, where there are fewer data points available for estimation.

To gain more insight into the results and the form of miscalibration, Figure 3.3 shows the predictive distribution over actions for the neural network as compared to the ground truth and approximate kNN models. The neural network model suffers from over-confident predictions (a result noted by Guo et al. [23]) and can also have very different predictions to what is expected; this indicates that it is unsuitable for use as a behaviour policy model. Note that the neural network achieved the highest held-out accuracy at predicting clinical action as compared to all other models (approximately 60%), but is still poorly calibrated: high accuracy does not imply good calibration.

Table 3.1 Average total variation distance between the estimated and target behaviour policy distributions for different policy models; logistic regression (LR), random forest (RF), neural network (NN), and approximate kNN; stratified by SOFA score (patient severity).

SOFA	LR	RF	NN	Approx kNN
0 - 4	0.249	0.214	0.213	**0.129**
5 - 9	0.269	0.254	0.246	**0.152**
10 - 13	0.309	0.309	0.399	**0.210**
14 - 23	0.356	0.337	0.426	**0.199**



(a) Overconfident predictions

(b) Incorrect predictions

Fig. 3.3 Examples of how neural networks can suffer from poor calibration as behaviour policy models. In Figure 3.3a, the neural network assigns higher probability to actions than expected; in Figure 3.3b the distribution of actions is very different to what is expected. The approximate kNN model does not suffer from these issues.

3.3 Approximate Model estimation

We now consider the task of building an approximate model (AM) of an MDP to directly estimate V^{π_e}. Correct AM estimation can be challenging in complex domains. Incorrect models can significantly worsen the performance of OPE estimators, so it is important to evaluate relevant methods and decide what is most appropriate. Several options were considered in this work:

Model the MDP directly: This involves learning a model for the transition distribution $P(s_{t+1}|s_t, a_t)$ and the reward function $R(s_t, a_t, s_{t+1})$ and then obtaining $\hat{V}^{\pi_e}(s)$ by averaging the returns from rollouts starting at s. This method is highly challenging for the sepsis domain. There is a significant degree of stochasticity from timestep-to-timestep in patient physiology, making it very difficult to predict the evolution of physiological features. The data available to us for learning the MDP model are also noisy and contain missing values (which have been imputed), complicating the prediction task. Developing accurate transition distribution models is therefore very difficult. The resulting degree of inaccuracy in transition distribution modelling (seen in Section 4.1) was unacceptable for evaluation purposes.

Fitted Q Iteration: This approach works directly in the continuous state-space and proceeds by solving a sequence of supervised learning problems that results in learning a parameterised function $f(s, a; \theta)$ that approximates $Q^{\pi_e}(s, a)$ [24]. At step $k + 1$ in the procedure, parameters θ_{k+1} are found by minimising the following loss function:

$$\theta^*_{k+1} = \arg\min_\theta \frac{1}{|\mathcal{D}|} \sum_{(s_t,a_t,r_t,s_{t+1})\in\mathcal{D}} \Big[\epsilon\left(\theta, \theta_k\right)\Big]^2$$

$$\epsilon\left(\theta, \theta_k\right) = r_t + \gamma \sum_{a'\in\mathcal{A}} \Big(\pi_e\left(a'|s_{t+1}\right) f\left(s_{t+1}, a'; \theta_k\right)\Big) - f(s_t, a_t; \theta)$$

We set $f(s, a; \theta_1) = 0$ at initialisation, and use $\gamma = 0.99$. We used a random forest with 80 trees to model $f(s, a; \theta)$, as this produced stable results. The scikit-learn library was used for implementation [18]. The specific loss function stated here incorporates a sum over actions – this is the form of the objective used in the Expected-SARSA algorithm [25], and led to reduced convergence time.

Kernel-based RL: Kernel-based methods, introduced in Ormoneit and Sen [26], can be used to learn $\hat{Q}^{\pi_e}(s, a)$. They have stability advantages over temporal difference methods such as Q-learning and SARSA learning. The implementation of kernel-based RL in this work followed the description in Ormoneit and Sen [26]. The radial basis function was used as the kernel.

Discretise the state-space and then use tabular SARSA learning: In this approach, the state-space was firstly discretised using k-means clustering. All states s_t were then assigned their nearest cluster centroid as their new label. Following this, the SARSA algorithm [27] was applied to learn $\hat{Q}^{\pi_e}(s, a)$. As the state-space was discretised, $\hat{Q}^{\pi_e}(s, a)$ was represented by a table of values, with entries for each state-action pair. This method was previously used by Komorowski et al. [10]. In this work, we used 1000 cluster centroids, which was found to provide a good tradeoff between convergence time and accuracy when running the algorithm.

We considered Fitted Q Iteration (FQI), Kernel-Based RL (KBRL) and Discrete SARSA when constructing approximate models.

3.4 Testing behaviour policy and approximate model estimation

3.4.1 OPE testing procedure

In the sepsis domain, we cannot easily verify the correctness of OPE because we do not have access to the ground truth value V^{π_e} for arbitrary π_e. To combat this problem, we proposed the following procedure:

1. Split the original dataset $\mathcal{D}$ into two subsets, $\mathcal{D}_1$ and $\mathcal{D}_2$.
2. Estimate the behaviour policy in $\mathcal{D}_1$. Let this be π_1.
3. Estimate the behaviour policy in $\mathcal{D}_2$. Let this be π_2.
4. Find the Monte Carlo estimate $\hat{V}^{\pi_1}_{\text{MC}}$ using $\mathcal{D}_1$, by averaging returns. This is an on-policy evaluation task and we have many trajectories, so we expect the estimate to be accurate. Therefore, assume $\hat{V}^{\pi_1}_{\text{MC}} \approx V^{\pi_1}$.
5. With π_2 as the behaviour policy and π_1 as the evaluation policy, use OPE (PHWIS, AM, or PHWDR) to find $\hat{V}^{\pi_1}_{\text{OPE}} = \hat{V}^{\pi_1}$ from $\mathcal{D}_2$.
6. Assess OPE quality by comparing $\hat{V}^{\pi_1}_{\text{MC}} \approx V^{\pi_1}$ and $\hat{V}^{\pi_1}_{\text{OPE}} = \hat{V}^{\pi_1}$. With good-quality OPE and correct behaviour/evaluation policy estimation, we expect good agreement with the Monte Carlo estimate.

We considered three different methods of splitting the trajectories – time based, random, and intervention based splitting:

- **Time:** The trajectories were divided into the two sets based on their starting times. We know the absolute time at which a patient's ICU record begins; we sorted these start times across all patients, and placed all trajectories starting before a threshold into $\mathcal{D}_1$, and all those after into $\mathcal{D}_2$. The policy followed by clinicians has changed slightly between older and newer trajectories; this might indicate a shift in clinical practices over time. The average total variation distance between π_1 and π_2 was about 0.15.
- **Random:** Randomly select half the trajectories to go in one set, and half to go in the other. In the limit of infinite data, the two behaviour policies will be identical, but in the finite data setting there are slight differences in the two policies. The average total variation distance between π_1 and π_2 was about 0.09.
- **Intervention:** Let $\mathcal{D}_2$ contain half of all the patients that did not receive any vasopressors (one of the two medications considered in the action-space) in their ICU stay and $\mathcal{D}_1$ contain all the other patients. The half of patients that were not treated with vasopressors that went into $\mathcal{D}_2$ were chosen randomly. The average total variation distance between π_1 and π_2 was about 0.29.

Random and intervention splitting used randomness when forming $\mathcal{D}_1$ and $\mathcal{D}_2$; more robust results were therefore obtained by averaging over at least 10 different behaviour/evaluation policy pairs. When computationally feasible, more pairs were used. With all methods of splitting, $\mathcal{D}_1$ and $\mathcal{D}_2$ were split further into training, validation, and testing sets (70:10:20).

We compared V^{π_1} and $\hat{V}^{\pi_1}$ using the mean squared error, $\text{MSE}(V^{\pi_1}, \hat{V}^{\pi_1})$. This was computed via a bootstrapped method. We repeated the following steps:

1. Sample $n = 200$ trajectories from $\mathcal{D}_2$.
2. Use OPE to get $\hat{V}^{\pi_1}$.
3. Repeat this process $k = 500$ times; each trial produces a sample of $\hat{V}^{\pi_1}$.
4. Compute the MSE between these samples and V^{π_1}.

3.4.2 Per-Horizon Weighted Importance Sampling (PHWIS) results

Table 3.2 presents the MSE when using the PHWIS estimator and different methods of splitting the dataset and behaviour policy estimation, using the methodology in

Section 3.4.1. The behaviour policy models are Approximate kNN (Approx kNN), Neural network (NN), Logistic Regression (LR) and Random Forest (RF).

On both the random and intervention splitting tasks, the approximate kNN model gives lower MSE than the neural network model, which may arise due to approximate kNN having superior calibration. Similar results are obtained on intervention-based splitting when comparing to RF. On the time-based splitting task, the approximate kNN model again has the best results. The RF and LR models obtain very poor results on this task; both models struggled to represent the behaviour policy well. The neural network does well on this task despite its calibration issues. However, it is unclear how robust this performance is, as only one behaviour/evaluation policy pair was considered. As the random and intervention based splitting experiments average across many different behaviour/evaluation policy pairs, they may be more reliable.

Table 3.2 MSE when using the PHWIS estimator with different methods of behaviour policy estimation to evaluate policies with time based, random, and intervention based splitting of the dataset. The behaviour policy models are Approximate kNN (Approx kNN), Neural Network (NN), Logistic Regression (LR) and Random Forest (RF).

	Approx kNN	NN	LR	RF
MSE time split	**3.35**	3.56	14.4	11.1
MSE random split	**2.48**	4.04	7.88	3.04
MSE intervention split	**2.04**	4.65	7.17	3.44

3.4.3 Approximate Model (AM) results

We now assess the performance of the AM estimators. Note that all AM estimation methods provide us with $\hat{Q}_M^{\pi_e}(s,a)$. We can evaluate a policy using the following relation (arising from the definition of $\hat{V}^{\pi_e}$), drawing n starting states from the evaluation dataset $\mathcal{D}_2$:

$$\hat{V}_{\mathrm{M}}^{\pi_e} = \frac{1}{n}\sum_{i=1}^{n}\sum_{a\in\mathcal{A}}\pi_e(a|s_0^{(i)})\hat{Q}_M^{\pi_e}(s_0^{(i)},a)$$

Table 3.3 shows the Mean Squared Error (MSE) from AM estimators using Fitted Q Iteration (FQI), Kernel-based RL (KBRL), and SARSA following discretisation (discrete SARSA). Due to the FQI method and discrete SARSA methods performing well on the time based splitting task, they were also examined for the other two splitting methods. The KBRL method appeared to have a stability issue for some tests and failed to converge properly; results are therefore not reported for KBRL on the random and intervention splitting tasks.

The FQI method performs well on the time based and random splitting tasks, and is superior to the other two methods. In both situations, the evaluation policy is relatively

close to the behaviour policy, making learning $\hat{Q}^{\pi_e}(s,a)$ straightforward. FQI uses $\mathcal{D}_2$, consisting of trajectories from π_b, to learn $\hat{Q}^{\pi_e}(s,a)$. When π_b and π_e are close, this problem is better conditioned. In intervention splitting, FQI does more poorly because the two policies are now more different, making this a harder learning task. Discrete SARSA has a similar issue on the intervention splitting task, but performs marginally better than FQI. However, the difference in performance is not very significant.

Table 3.3 MSE when using three different AM approaches for OPE.

	Fitted Q Iteration	Discrete SARSA	Kernel-based
MSE time split	**0.0622**	1.64	2.37
MSE random split	**0.177**	1.45	—
MSE intervention split	3.87	**3.77**	—

3.4.4 Per-Horizon Weighted Doubly Robust (PHWDR) results

Table 3.4 shows the MSE when using the PHWDR estimator with different approximate model terms and different methods of behaviour policy estimation. The behaviour policies are Approximate kNN (kNN) and Neural Network (NN). The AM methods are Fitted Q Iteration (FQI), Discrete SARSA (SARSA) and Kernel-based RL (KBRL).

The FQI approximate model terms help to reduce the MSE in time based and random splitting, as compared to using IS alone (i.e. with the PHWIS estimator in Table 3.2), for both the approximate kNN and neural network behaviour policy models. As seen in Table 3.3, the MSE of the AM in both these situations is very low, so the AM terms improve the quality of the estimate. The difference in performance due to better behaviour policy modelling is not as notable – performance is dominated by the AM terms. Therefore, although that the neural network is poorly calibrated, the MSE obtained using it as a behaviour policy model in PHWDR is slightly lower than the MSE when using approximate kNN. This difference is, however, very small (0.02).

In the intervention based splitting scenario, the MSE is high because the FQI AM performs poorly, and hence worsens the estimates (regardless of the behaviour policy model). Once again, the AM terms dominate. Interestingly, although the SARSA AM had lower MSE than the FQI AM in intervention splitting (as shown in Table 3.3), when used in PHWDR, the FQI AM performs better. This could be due to the fact that the AM estimator uses only the the action value estimate for the first state in a trajectory, $\hat{Q}_M^{\pi_e}(s_0,a)$, but PHWDR requires all $\hat{Q}^{\pi_e}$ values. SARSA may represent the initial timestep more accurately, but fails to represent the others as well. For this reason, the FQI model is preferable for use in PHWDR.

These results show how poor quality model terms can worsen the quality of the evaluation result when using PHWDR, as seen when using approximate kNN and KBRL/SARSA, and the neural network and KBRL/SARSA. Careful analysis when deciding the appropriate model classes to use for AM estimation is therefore important.

Table 3.4 MSE when using the PHWDR estimator with different behaviour policy models and approximate model methods.

	kNN-FQI	NN-FQI	kNN-SARSA	NN-SARSA	kNN-KBRL	NN-KBRL
MSE time split	**3.05**	3.11	6.53	5.80	6.54	10.2
MSE random split	2.04	**2.02**	10.21	9.17	—	—
MSE intervention split	**3.90**	**3.90**	4.26	9.15	—	—

3.4.5 Conclusions

To summarise the results:

- **Behaviour policy estimation:** Approximate kNN is the most appropriate behaviour policy model, given its well-calibrated nature.
- **Approximate model estimation:** FQI often performs the best, or close to the best. In one situation, discrete SARSA obtained superior performance, but the difference between discrete SARSA and FQI's performance was small.
- **Per-Horizon Weighted Doubly Robust (PHWDR) estimator:** The combination of approximate kNN and FQI was found to perform well. Discrete SARSA did not work well as an AM in PHWDR. Although the neural network behaviour policy model obtained comparable results to approximate kNN when used in PHWDR with FQI as the AM, this was mainly because the AM dominated the behaviour policy. Given that approximate kNN is better calibrated, it is the preferred choice.

Chapter 4

Model-Based Reinforcement Learning

This chapter explores the use of model-based reinforcement learning (RL) to learn medical treatment strategies for patients with sepsis, and considers the construction of effective environment models, the use of environment models to learn policies, and an evaluation of the learned policies.

4.1 Environment models

The first stage in model-based RL is to learn a model for the transition distribution, P, of the underlying Markov Decision Process (MDP).

The task of learning a transition distribution can be equivalently described as the task of learning an environment model. This framing of the problem is motivated by the discussion in Nagabandi et al. [28].

The objective is to predict $\Delta_t = s_{t+1} - s_t$, the change in a patient's physiological state, which is achieved by learning a function $f(h_t; \theta)$ where:

$$\Delta_t = f(h_t; \theta) + \epsilon; \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$
$$h_t = g\left(s_t, a_t, s_{t-1}, a_{t-1}, \dots\right)$$

In this work, when used, the function g concatenates the current state-action pair (s_t, a_t) with the previous three timesteps' state-action pairs. This is to capture some amount of historical information about the patient's physiology. Cross-validation performance (in terms of prediction mean squared error) motivated the use of three past timesteps of information.

We now consider the different approaches used to construct the environment model.

4.1.1 Constructing the environment model

To learn the environment model, represented by the function f, several approaches were tried, including Non-Bayesian models (linear regression, feedforward neural networks, recurrent neural networks) and Bayesian models (Bayesian neural networks). These are described in more detail here.

Linear Regression: These models take in the adjusted state vector h_t and apply a linear transformation: $\widehat{\Delta}_t = Wh_t + b$, where W is a weight matrix and b is a bias vector, both of which are learned by minimising the mean squared error between $\widehat{\Delta}_t$ and Δ_t, via gradient descent:

$$\theta^* = \arg\min_\theta \frac{1}{|\mathcal{D}|} \sum_{(s_t, a_t, s_{t+1}) \in \mathcal{D}} \|(\Delta_t - \widehat{\Delta}_t)\|^2$$

This was implemented using the scikit-learn library [18].

Feedforward Neural Networks: These models take as input h_t and apply a series of parameterised linear transformations (with weight matrices W_i and biases b_i, with i indexing the layer) and pointwise nonlinearities (σ_i) to produce an output. For example, in the case of a two-layer network: $\widehat{\Delta}_t = \sigma_2 \left(W_2 \ \sigma_1 \left(W_1 h_t + b_1\right) + b_2\right)$.

Several different network architectures were experimented with. The best performing architecture had the following design:

- Input: h_t
- Fully-connected layer, 128 units, ReLU nonlinearity
- Batch-normalization [29]
- Fully-connected layer, 32 hidden units, ReLU nonlinearity
- Batch-normalization
- Fully-connected layer, linear activation (producing the output $\widehat{\Delta}_t$)

Note that the ReLU nonlinearity is described by: $\sigma(z) = \max(0, z)$

This network was trained using stochastic gradient descent (SGD) and the Adam [22] optimiser. This was implemented using TensorFlow [21].

Recurrent Neural Networks (RNNs): These are temporal models that use some embedding of history, encapsulated in a hidden state, and the current input to make

predictions. A simple RNN can be described as follows:

$$c_t = \sigma_h(W_x x_t + W_c c_{t-1} + b_c)$$
$$y_t = \sigma_y(W_y c_t + b_y)$$

where c_t is the hidden state at time t, and y_t is the output at time t. W_i and b_i are trainable weight matrices and bias vectors respectively. Note that c is used for the hidden state and not h, as is conventional, because h is used to describe the output of the concatentation function $g(\cdot)$

For environment modelling, a more powerful RNN model, an LSTM [30], was used. The output y_t is now $\widehat{\Delta}_t$. As with the other models, the model was trained to minimise the mean squared error between $\widehat{\Delta}_t$ and Δ_t. The model was trained using stochastic gradient descent (SGD) and the Adam [22] optimiser. As the RNN computes its own representation for the historical state, at every timestep, the raw state and action vector is passed in — that is, (s_t, a_t), rather than h_t. RNNs were implemented using TensorFlow [21].

Bayesian Neural Networks (BNNs): These models represent the full predictive distribution over $\widehat{\Delta}_t$ instead of providing point estimates. The description provided here is based on that in Depeweg et al. [31].

As with non-Bayesian neural networks, a likelihood model is specified, mapping the input to the output, using a series of linear transformations and pointwise nonlinearities. A prior distribution is specified over the values of the parameters of the weight matrices and bias vectors; in this case, it is a Gaussian with zero mean and specifiable variance: $\theta_i \sim \mathcal{N}(0, \sigma_i^2)$.

In BNN modelling, we aim to form the posterior distribution over the parameters given the training data, consisting of (h_t, Δ_t) pairs, and marginalise out the parameters to get the predictive distribution over $\widehat{\Delta}_t$.

However, as this marginalisation procedure is intractable, we form a variational approximation to the posterior distribution, $q(\theta|\mathcal{D})$, and minimise the alpha-divergence between the true posterior and the variational approximation [32]. This minimisation procedure aims to find the parameters of the approximate posterior; that is, the mean and variance of the Gaussian distributions that make up the approximate posterior distribution.

Samples from the predictive distribution are then obtained by sampling parameter values from the approximate posterior and averaging:

$$p(\widehat{\Delta}_t|h_t,\mathcal{D}) = \int p(\widehat{\Delta}_t|h_t,\mathcal{D},\theta)q(\theta|\mathcal{D})d\theta$$
$$\approx \frac{1}{N}\sum_{i=1}^{N} p(\widehat{\Delta}_t|h_t,\mathcal{D},\theta_i); \quad \theta_i \sim q(\theta|\mathcal{D})$$

The best BNN had two hidden layers, each with 32 hidden units and tanh nonlinearities. It was implemented using the Autograd library [33].

As preprocessing, the input data were standardised to zero mean and unit variance. Similar processing was applied to the target output values.

4.1.2 Environment modelling results

Table 4.1 shows the mean squared error (MSE) when predicting Δ_t when using different environment models: Linear Regression (LR), feedforward Neural Network (NN), Recurrent Neural Network (RNN), and Bayesian Neural Network (BNN). Non-Bayesian models appear to perform better than Bayesian models; in part, this is due to the difficulty in tuning hyperparameters of Bayesian neural networks.

Table 4.1 Mean squared error (MSE) on a held-out validation set when predicting Δ_t for different environment models: Linear Regression (LR), feedforward Neural Network (NN), Recurrent Neural Network (RNN), and Bayesian Neural Network (BNN).

	LR	NN	RNN	BNN
MSE	0.195	0.171	**0.122**	0.220

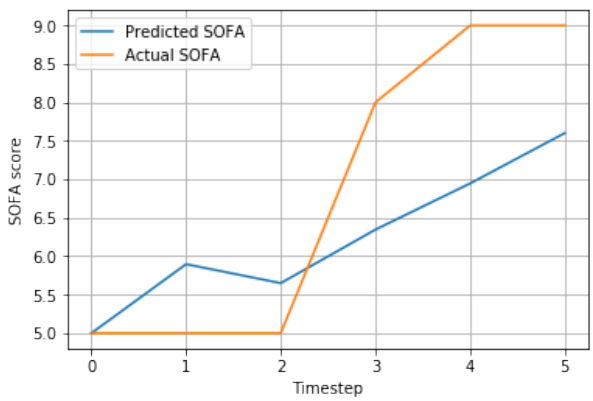
The RNN obtains the best MSE by this metric. Analysis of the performance reveals that this low MSE is obtained because the predictions produced by the RNN at large timesteps are very accurate. The RNN likely performs well at a large value of t because for all $t' \leq t$, the RNN has taken in the ground truth values of $s_{t'}$ as input, and therefore can form a meaningful representation of the entire history of the patient, which enables it to predict Δ_t accurately. The other models take only the last three states in raw form as input, which makes predicting Δ_t more challenging. Passing in the entire history to the other models would involve processing a roughly 1000-dimensional input vector at every timestep in each model, which presents computational challenges. However, the performance of the RNN at small values of t are poor, as compared to the other models; the RNN does not seem to have the capacity to predict Δ_t accurately at small t. Increasing the modelling power via adding additional layers or more hidden units causes severe overfitting, which regularisation methods (such as weight decay) are not

able to address. Due to this reason, the feedforward neural network is preferred as an environment model.

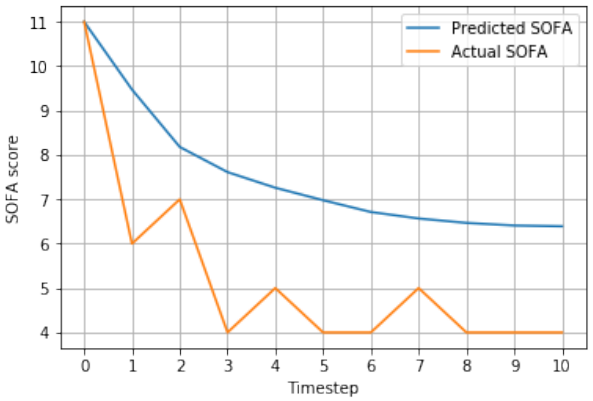
Figure 4.1 shows some examples of rollouts for the SOFA score, a measure of patient severity, using the feedforward neural network environment model. To generate these plots, a trajectory was randomly sampled from a validation set. The initial state in this trajectory, with the concatenated history (zeroed out initially as there is no history for the first state), h_1 was passed as input to the network, producing $\widehat{\Delta}_1$. The predicted next state was calculated using: $\hat{s}_2 = s_1 + \widehat{\Delta}_1$. $\hat{h}_2$ is computed from this by the concatenation process, and using the clinician action at $t = 2$: a_2. We then use $\hat{h}_2$ to calculate $\widehat{\Delta}_2$. Note that we only use the true state to initialise the process; after that, we use the model's predictions, making this an accurate representation of the use of the environment model as an (approximate) simulator.

These plots show that neural network sometimes can represent the overall trend in SOFA score over a patient's trajectory accurately; this is most noticeable for Figures 4.1a, 4.1b and 4.1c. However, the task of environment modelling is clearly very challenging – there are large changes in the value of the SOFA score at certain timesteps, and there are several timesteps where the values do not change at all (likely to there being missing data at these points which have been imputed). The accuracy of the predictions worsens as we increase the length of the rollout; this is understandable, as prediction errors compound over multiple timesteps.

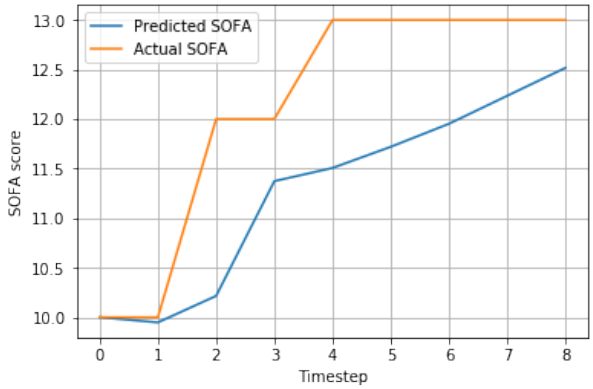
Although these plots illustrate that the modelling performance is not highly accurate, given that the neural network environment model is able to capture the overall trend of the SOFA score in some examples, it may offer sufficient performance to be able to assist in discovering improved policies, which is explored in the next section. It is clear however that there is more investigation required into the best choice of environment model for this domain.



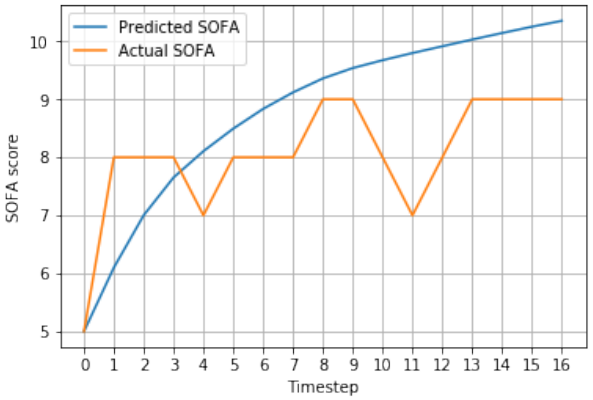
(a) Rollout 1



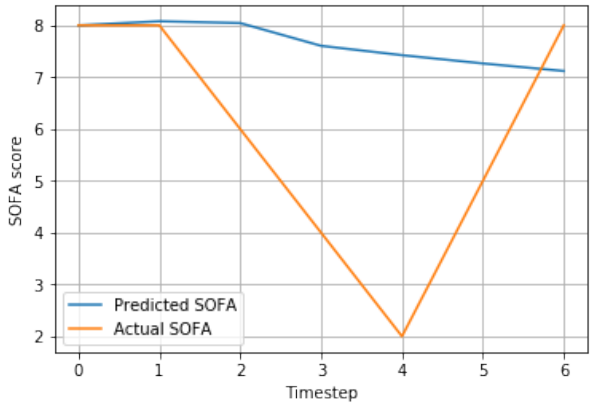
(b) Rollout 2



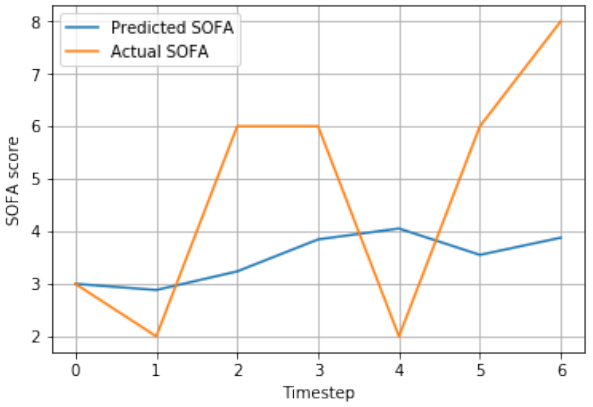
(c) Rollout 3



(d) Rollout 4



(e) Rollout 5



(f) Rollout 6

Fig. 4.1 Examples of rollouts from the neural network environment model for the SOFA score, an important physiological feature that captures severity of sepsis. Predictions from the network can approximately capture the trend in SOFA score in certain cases, but not others. The irregular fluctuations exhibited by the true SOFA score illustrate the difficulty of the task.

4.2 Policy search

After learning an environment model, policy search algorithms are used to find an appropriate policy π. This section details the approach used to discover a suitable policy.

4.2.1 Procedure for policy discovery

In medical settings, it is important not to stray too far from the policy used by clinicians for interpretability reasons; therefore, the approach to discover improved treatment strategies proceeded as follows:

- Learn the parameters ϕ describing the policy followed by clinicians, $\mu(a|h_t;\phi)$, using supervised learning.
- Initialise the parameters θ of a new policy $\pi(a|h_t;\theta)$ using ϕ.
- Using the learned environment model, simulate rollouts from $\pi(a|h_t;\theta)$. Use a policy improvement algorithm with a small learning rate (to ensure small policy updates, staying close to μ) to improve π.

By initialising π to be similar to the treatment strategy followed by clinicians and only allowing small updates (via the small learning rate and limited training steps) we can find a final treatment strategy that, although not necessarily optimal (as it is based on what clinicians follow), improves on what doctors currently practice and remains interpretable.

Modelling clinician behaviour: The first step of this procedure is to model clinician behaviour by learning the policy μ. We formulated this as a classification task, where we learned parameters ϕ^* that minimised the following objective:

$$\phi^* = \arg\min_\phi \frac{1}{|\mathcal{D}|} \sum_{(s_t,a_t)\in\mathcal{D}} \sum_{a'\in\mathcal{A}} -\mathbb{1}(a_t = a') \log\Big(\mu(a'|h_t;\phi)\Big)$$

This objective function is the average cross-entropy between the predictive distribution over actions from the neural network and the label distribution (encoding the action taken by the clinician).

The model that performed well on this task was a two-layer feedforward neural network with 64 hidden units in each layer, with ReLU activations. An L2 regularisation penalty was added to the loss to prevent overfitting. This was implemented using TensorFlow [21]. The resulting held-out accuracy in predicting actions was about 60%; the fact this is low represents how clinicians are quite stochastic in their decision making when treating sepsis.

Policy improvement: Having modelled the clinician's policy, we then improved it using the environment model. We used policy improvement algorithms, with the environment model acting as a 'simulator' for us to sample trajectories, or rollouts, from. In this process, we sampled a random trajectory from the dataset, and selected the starting state. We then performed a rollout from this starting state for 10 timesteps (as discussed, long horizons suffer from very poor predictions, so this was a compromise between the typical length of a trajectory in the dataset and the quality of the rollout), using actions sampled from $\pi(a|h_t;\theta)$. We recorded the states, actions, and rewards, and then used gradient ascent with these sampled trajectories to improve the policy.

Two policy improvement algorithms were compared – the REINFORCE Policy Gradient algorithm (PG) [34], which is a high variance, yet simple, method, and the Proximal Policy Optimization algorithm (PPO) [35], which is a lower variance actor-critic method employing an approximate trust-region optimisation procedure to ensure stable updates to π. The PPO algorithm has additional modelling complexity, as we must simultaneously learn a critic to model $V^{\pi}(s)$; for this reason it is informative to compare PPO to a baseline. The critic was modelled using a single layer neural network with 100 hidden units and ReLU activation. All procedures were implemented in TensorFlow [21].

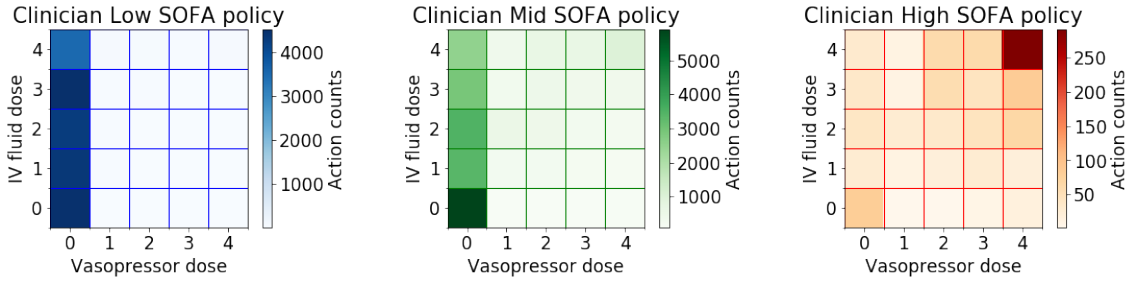
4.3 Qualitative analysis of learned policies

Qualitative results for the performance of the models in different severity regimes are presented here; that is, timesteps at which patients had relatively low SOFA scores (under 5), timesteps at which patients had medium SOFA scores (5 – 15) and timesteps at which patients had high SOFA scores (over 15). This is to understand how the model performs on different severity subcohorts.

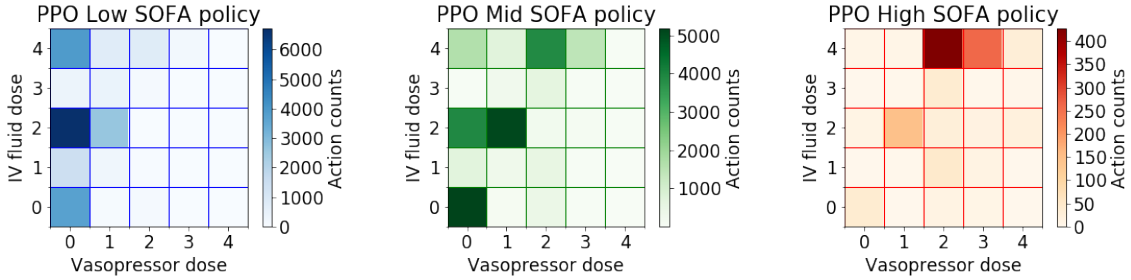
Figure 4.2 shows the discovered treatment policies on a held-out test set when using the PPO algorithm and the PG algorithm. Also included is the policy used by the clinician. The action numbers index the different discrete actions selected at a given timestep, and the charts shown aggregate actions taken over all timesteps for those cohorts. Action 0 refers to no drugs given to the patient at that timestep, and increasing actions refer to higher drug dosages, where drug dosages are represented by quartiles. As shown, physicians do not often prescribe vasopressors to patients, unless patients have very high SOFA scores (note the high density of actions corresponding to zero vasopressor dose in the first two histograms) and this behavior is reflected in the policy learned by both the PPO and PG models. This result is clinically interpretable; even though vasopressors are commonly used in the ICU to elevate mean arterial pressure, many patients with sepsis do not have low blood pressure and therefore do not need

vasopressors. Additionally, there have not been many clinical trials reporting improved patient outcomes from using vasopressors [36].

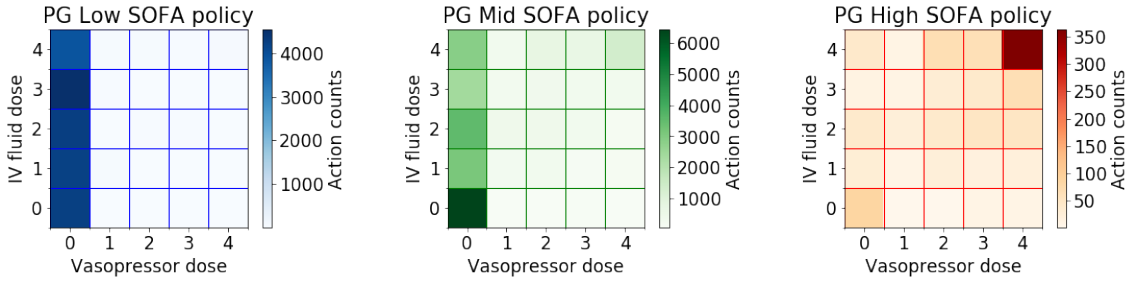
The PG algorithm's policy remains very close to that used by clinicians when considering aggregated actions. The reason for this is likely the small learning rate used. With a larger learning rate, the policy deviates quite significantly from that followed by clinicians and is no longer clinically interpretable. The PPO algorithm's policy has some similarities to what clinicians follow, but also exhibits some noticeable differences, especially in the middling-severity regime. Of note is the high density of actions with a middling vasopressor and high IV fluid dose: action index (2,4). This result is not observed in the clinician's policy.



(a) Clinician treatment policy



(b) PPO treatment policy



(c) Policy Gradient treatment policy

Fig. 4.2 Treatment policy followed by the clinician and those discovered by the models on a held-out test set, stratified by patient severity.

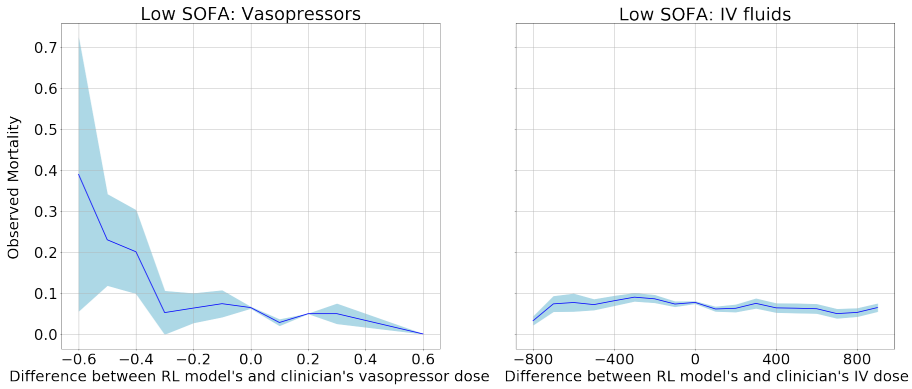
Figure 4.3 shows the correlation between the observed mortality, and the difference between the treatment doses suggested by the PPO algorithm's policy and the actual doses given by clinicians. The dosage differences at individual timesteps were binned, and mortality counts were aggregated. The mean and standard deviation of the observed

mortality proportion in each bin are shown. The results are stratified based on patient SOFA score. Figure 4.4 depicts the same plot for the policy learned by the PG algorithm.

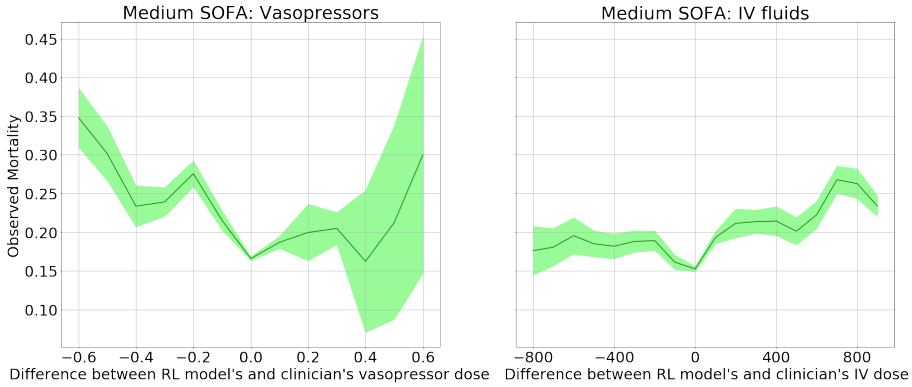
From this qualitative analysis, it appears that the PPO algorithm learns a strong treatment policy in the medium SOFA regime. We observe low mortalities when the dosage from the learned policy and dosage given by clinicians coincide, at a difference of 0, indicating the potential validity of the learned policy; when clinicians act according to what is suggested by the learned policy, patients appear to have better outcomes. The observed mortality proportion then increases as the difference between the optimal dosage and the true dosage increases. The PG algorithm's policy does not show this particular characteristic.

In the low SOFA regime, the baseline mortality is quite low; as a result, these qualitative plots are not quite as informative.

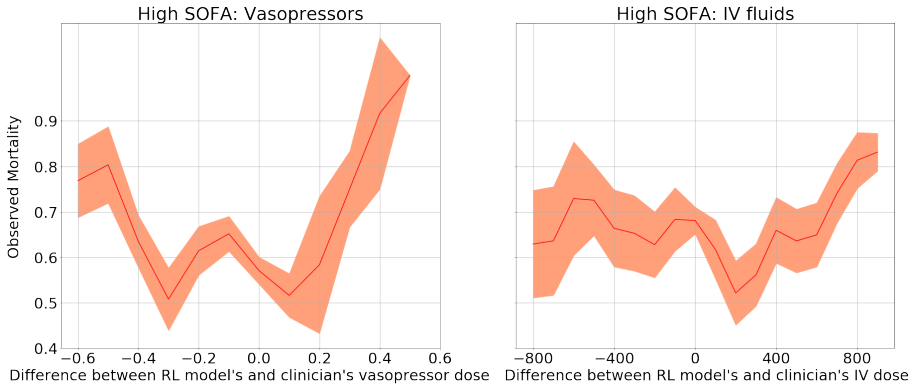
For high SOFA scores, the performance of both models appears to be weaker, with zero deviation not indicating the best outcomes. This could be due to the lack of data points available for this regime, and also the difficulty in learning a strong policy when patients have severe sepsis. It is likely that the environment model in this regime is inaccurate, as patients with severe sepsis exhibit significant physiological changes from timestep to timestep, which the environment model is not able to represent effectively. It may be reasonable to follow the treatment policy suggested by clinicians when patients have high severity sepsis – this is explored further in Section 4.4.



(a) PPO low SOFA

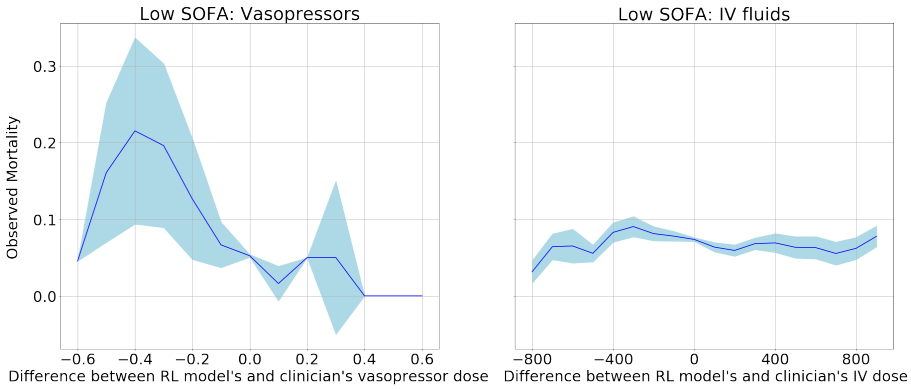


(b) PPO mid SOFA

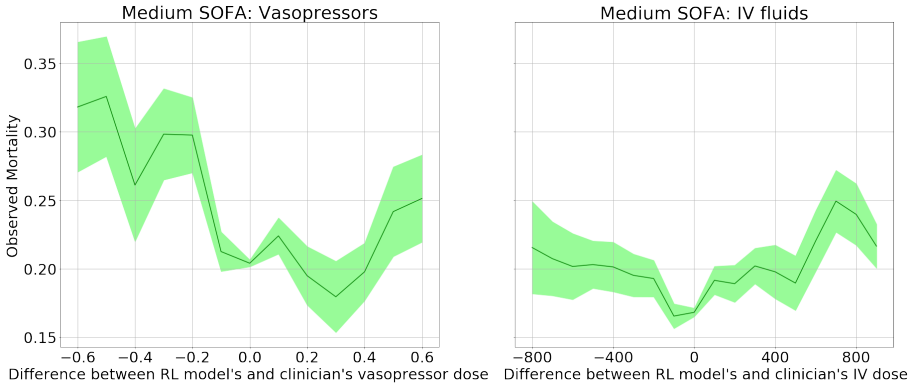


(c) PPO high SOFA

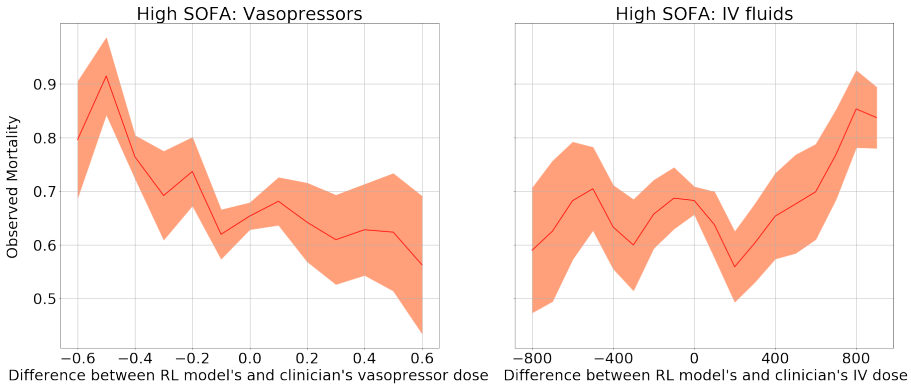
Fig. 4.3 Comparison of how observed mortality varies with the difference between the dosages recommended by the learned policy using the PPO algorithm and the dosages administered by clinicians on a held-out test set.



(a) PG low SOFA



(b) PG mid SOFA



(c) PG high SOFA

Fig. 4.4 Comparison of how observed mortality varies with the difference between the dosages recommended by the learned policy using the PG algorithm and the dosages administered by clinicians on a held-out test set.

4.4 Quantitative analysis of learned policies

This section considers the quantitative evaluation of polices learned using model-based RL, supplementing the qualitative analysis in Section 4.3. We firstly consider the performance using the PHWIS estimator.

4.4.1 Results using the PHWIS estimator

Table 4.2 shows OPE results for the best configurations of the Proximal Policy Optimisation (PPO) and REINFORCE Policy Gradient (PG) algorithms, on a held out test set. In order to form the estimates, an approximate k-Nearest Neighbours (kNN) model was used to represent the behaviour policy, using 250 neighbours to form the distribution estimate. The PHWIS estimator is used to generate these results as it does not require learning an approximate model (AM) of the MDP, and so is useful to consider for initial analysis. Note that the value of the clinician's policy, V^{π_b}, obtained using the Monte Carlo estimator, is found to be 9.90.

Table 4.2 OPE results using the PHWIS estimator for the best configurations of the two policy improvement algorithms considered – Proximal Policy Optimisation (PPO) and the REINFORCE Policy Gradient algorithm (PG). The PPO algorithm achieves an improvement over the clinician policy (clinician value estimate using Monte-Carlo is 9.90), whereas the PG algorithm only improves slightly on clinician performance.

	PPO (90 training iterations)	PG (30 training iterations)
$\hat{V}^{\pi_e}_{\text{PHWIS}}$	**10.8**	9.93

The best-performing PPO algorithm performs better than the clinician policy, but not by a significant amount. The PG algorithm does not appear to do as well in this test, and only just outperforms the clinician. This reinforces the conclusions drawn in Section 4.3, where the qualitative analysis indicated that the PG algorithm did not improve on what clinicians followed. The reasoning for this could be high variance of the specific PG method used (the REINFORCE algorithm); the PPO algorithm, through using an actor-critic method, does not suffer as seriously from this issue.

4.4.2 Blending clinical and discovered treatment policies

Section 4.3 revealed that when patient states were stratified by SOFA score, the PPO algorithm showed improvement over the strategy followed by clinicians in the medium SOFA regime, but did not learn as effective policies in the low SOFA and high SOFA regime, due to the lack of clear signal in the low SOFA regime (low baseline mortality), and the lack of data points in the high SOFA regime (leading to poorer environment model performance). Given the safety-critical nature of this domain, it is well-motivated

to consider relying on the clinician's policy in regimes where the model is believed to have inferior performance.

Table 4.3 considers the best-performing PPO model from Section 4.4.1 and presents results using the PHWIS, PHWDR, and Approximate Model (AM) estimators when we adjust what policy is active in each severity regime (clinician or model-based RL). To obtain the AM estimator and the AM terms for the PHWDR estimator, the Fitted Q Iteration (FQI) algorithm [24] was used with a random forest with 80 trees to represent the quantity $\hat{Q}^{\pi_e}(s, a)$.

Firstly, these results show that the AM estimator does not appear to discriminate significantly between the learned policies; all policies have similar values. The value of the clinician policy from FQI, $\hat{V}^{\pi_b}_{\text{AM}}$, is found to be 9.36, again very similar to the values of the other evaluation policies. It is therefore difficult to draw conclusions from considering these results in isolation.

When considering the PHWIS and PHWDR estimators, it appears that the best policy is obtained when relying on the clinical policy in the low and high severity regimes, and the RL-learned policy in the medium severity regime. The two estimation procedures show reasonable agreement in estimated value, though there are discrepancies between them. However, as both show similar trends, and broadly agree with the conclusions found from the qualitative analysis (using the learned policy in the medium severity regime can offer improvements), it may be possible to conclude that this policy does improve on what clinicians currently follow. Investigating the specific policy discovered in this regime could offer valuable clinical insight.

The fact that this particular blending strategy obtains good results can be explained. It is likely that in low/high severity regimes, clinicians have a set protocol they follow, which performs relatively well; it therefore does not make sense to deviate from this. This is clearly seen in the high severity regime where clinicians often prescribe the maximum dosage of both drugs considered. In the medium severity regime, there is more variability in clinician action, and model-based RL is effective here in identifying suitable courses of action.

An extension on this analysis is to devise the most appropriate blending strategy when combining the clinician and PPO policies; the method used here is quite coarse, and it is reasonable to expect further improvements if the blending strategy is refined.

Table 4.3 OPE results using the PHWIS, PHWDR, and AM estimators when relying on combinations of the clinician's policy and the PPO policy. As a comparison, results are also shown for following the clinical policy in all regimes. By relying on the clinician policy in regimes where the PPO policy is understood to not perform as effectively, improvements in performance are observed for the PHWIS and PHWDR estimators.

Low SOFA	Medium SOFA	High SOFA	$\hat{V}^{\pi_e}_{\text{PHWIS}}$	$\hat{V}^{\pi_e}_{\text{PHWDR}}$	$\hat{V}^{\pi_e}_{\text{AM}}$
PPO	PPO	PPO	10.8	11.5	9.33
PPO	PPO	Clinician	11.7	11.8	9.35
Clinician	PPO	Clinician	**12.1**	**12.8**	9.35
PPO	Clinician	PPO	7.63	7.58	9.34
Clinician	Clinician	Clinician	10.2	9.87	**9.36**

Chapter 5

Conclusion

5.1 Main findings

This work explored the use of reinforcement learning (RL) to provide medical decision support for sepsis treatment. The project examined two areas: developing an empirical evaluation procedure to assess the quality of sepsis treatment strategies, and investigating the use of model-based RL algorithms to deduce improved treatment strategies for sepsis.

The empirical evaluation procedure developed in this work addressed two notable challenges with using existing Off-Policy Evaluation (OPE) methodologies in observational studies – how to estimate an unknown behaviour policy and approximate model (AM) terms for use in OPE. The importance of well-calibrated behaviour policy models was discussed, and a non-parametric method based on approximate k-Nearest Neighbours (kNN) was found to give superior results over parameteric models such as random forests and neural networks. A procedure for estimating AM terms using random forests and Fitted Q Iteration (FQI) [24] gave promising results, over alternative methods using kernel-based reinforcement learning [26] and temporal-difference SARSA learning. OPE estimators using approximate kNN and FQI were found to give the best performance when evaluating treatment strategies for sepsis.

The investigation into model-based RL revealed that neural networks can be used to model the physiological dynamics of patients with sepsis (they can be used as environment models). Although the task of modelling physiology is challenging, these environment models appear to be accurate enough for use in discovering improved treatment strategies. When using policy search procedures to learn medical treatment strategies for sepsis, an actor-critic algorithm, Proximal Policy Optimisation (PPO) [35], demonstrated encouraging qualitative results. The policy search approach uses the treatment strategy followed by clinicians as an initialisation point in order to ensure that the treatment strategy learned after using PPO is still clinically interpretable.

The developed evaluation procedure, with an approximate kNN behaviour policy model and an FQI AM, was used to assess the treatment strategies learned with model-based RL. Overall, this treatment strategy appeared to improve on that followed by clinicians, and performed well for patients that had medium severity sepsis. However, it performed more poorly when patients had low/high severity sepsis. This result was corroborated by the qualitative analysis. By blending the strategy used by clinicians and that discovered by RL, a potentially more effective policy was found. This is an encouraging result, showing that the combination of existing clinical intuition and use of observational data could help improve the standard of care for patients with sepsis.

5.2 Meeting of objectives

The initial objectives are restated here:

1. Develop an empirical evaluation procedure to assess the quality of medical treatment strategies for sepsis.

2. Investigate whether continuous state-space model-based RL algorithms can be used to find improved treatment policies for sepsis.

This study has contributed important results towards both these objectives.

The developed evaluation procedure identified the key issue of *calibration* for behaviour policy modelling, and proposed an estimation approach to ensure good calibration, namely k-Nearest Neighbours. The procedure also discovered the most appropriate model class/learning algorithm to represent approximate model terms – Fitted Q Iteration. OPE estimators combining these methods were found overall to give the best results. This work also formulated the step-wise Per-Decision Weighted Importance Sampling (step-wise PHWIS) and Per-Decision Weighted Doubly Robust (PHWDR) estimators, extending prior work.

In terms of model-based RL, this work investigated different methods of environment modelling, and found feedforward neural networks to be the most appropriate technique. The proposed procedure for policy search (discovering improved medical treatment policies) using the Proximal Policy Optimization [35] algorithm allows learned treatment strategies to remain interpretable and to potentially improve on what clinicians currently follow. This was verified using qualitative analysis and different methods of quantitative analysis.

5.3 Further work

5.3.1 Off-Policy Evaluation

An important direction of future work is to examine whether the OPE procedure developed here works well for other medical problems; for example, mechanical ventilation weaning [37], and optimal heparin dosing [38].

It is also of interest to develop an evaluation procedure that avoids importance sampling (IS), for several reasons. Firstly, using IS in OPE can result in high variance estimators, which is undesirable. Secondly, avoiding IS may prevent the need to model the behaviour policy, which could improve the quality of OPE, as poorly estimated behaviour policies can lead to unreliable evaluation. Thirdly, IS approaches are not effective when evaluation policies are deterministic. If the evaluation policy is deterministic, the importance weight will become zero unless the behaviour policy and evaluation policy agree on the action taken on every timestep. For long trajectories, this agreement is highly unlikely. An approach without IS may be able to address this challenge.

5.3.2 Model-based RL

Learning high-quality policies with model-based RL is most likely limited by the efficacy of the developed environment model. Building a more accurate environment model could lead to discovering superior treatment strategies. For example, the use of domain knowledge in the construction of the environment model could allow physiological dynamics to be captured more effectively.

More work could also be done on blending clinician and model-based RL treatment strategies. It is also of interest to understand when clinicians make mistakes/act suboptimally, which this direction of investigation may reveal.

References

[1] J. Cohen, J.-L. Vincent, N. K. J. Adhikari, F. R. Machado, D. C. Angus, T. Calandra, K. Jaton, S. Giulieri, J.Delaloye, S. Opal, K. Tracey, T. van der Poll, and E. Pelfrene, "Sepsis: a roadmap for future research," *Lancet Infectious Diseases*, vol. 15, no. 5, p. 581–614, 2006.

[2] J.-L. Vincent, Y. Sakr, C. L. Sprung, V. M. Ranieri, K. Reinhart, H. Gerlach, R. Moreno, J. Carlet, J.-R. L. Gall, and D. Payen, "Sepsis in European intensive care units: results of the SOAP study," *Critical Care Medicine*, vol. 34, no. 2, pp. 344–353, 2006.

[3] J. Waechter, A. Kumar, S. E. Lapinsky, J. Marshall, P. Dodek, Y. Arabi, J. E. Parrillo, R. P. Dellinger, A. Garland, and Cooperative Antimicrobial Therapy of Septic Shock Database Research Group and others, "Interaction between fluids and vasoactive agents on mortality in septic shock: a multicenter, observational study," *Critical care medicine*, vol. 42, no. 10, pp. 2158–2168, 2014.

[4] A. Rhodes, L. E. Evans, W. Alhazzani, M. M. Levy, M. Antonelli, R. Ferrer, A. Kumar, J. E. Sevransky, C. L. Sprung, M. E. Nunnally *et al.*, "Surviving sepsis campaign: International guidelines for management of sepsis and septic shock: 2016," *Intensive care medicine*, vol. 43, no. 3, pp. 304–377, 2017.

[5] A. Raghu, M. Komorowski, L. A. Celi, P. Szolovits, and M. Ghassemi, "Continuous state-space models for optimal sepsis treatment: a deep reinforcement learning approach," in *Machine Learning for Healthcare Conference*, 2017, pp. 147–163.

[6] D. Precup, R. S. Sutton, and S. P. Singh, "Eligibility traces for off-policy policy evaluation," in *International Conference on Machine Learning*, 2000, pp. 759–766.

[7] N. Jiang and L. Li, "Doubly robust off-policy value evaluation for reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 652–661.

[8] P. Thomas and E. Brunskill, "Data-efficient off-policy policy evaluation for reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 2139–2148.

[9] M. Farajtabar, Y. Chow, and M. Ghavamzadeh, "More robust doubly robust off-policy evaluation," *CoRR*, vol. abs/1802.03493, 2018.

[10] M. Komorowski, A. Gordon, L. A. Celi, and A. Faisal, "A Markov Decision Process to suggest optimal treatment of severe infections in intensive care," in *Neural Information Processing Systems Workshop on Machine Learning for Health*, December 2016.

[11] V. Pong, S. Gu, M. Dalal, and S. Levine, "Temporal difference models: Model-free deep RL for model-based control," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=Skw0n-W0Z

[12] S. Doroudi, P. S. Thomas, and E. Brunskill, "Importance sampling for fair policy selection," 2017.

[13] A. E. W. Johnson, T. J. Pollard, L. Shen, L.-W. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "MIMIC-III, a freely accessible critical care database," *Scientific Data*, vol. 4, no. 160035, p. 122, 2016.

[14] M. Singer, C. S. Deutschman, C. Seymour *et al.*, "The third international consensus definitions for sepsis and septic shock (sepsis-3)," *JAMA*, vol. 315, no. 8, pp. 801–810, 2016. [Online]. Available: http://dx.doi.org/10.1001/jama.2016.0287

[15] R. Sutton and A. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[16] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, W. D, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[17] P. E. Marik, W. T. Linde-Zwirble, E. A. Bittner, J. Sahatjian, and D. Hansell, "Fluid administration in severe sepsis and septic shock, patterns and outcomes: an analysis of a large national database," *Intensive care medicine*, vol. 43, no. 5, pp. 625–632, 2017.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 1998, pp. 604–613.

[20] E. Bernhardsson, "ANNOY: Approximate nearest neighbors in C++/Python optimized for memory usage and loading/saving to disk," 2016, accessed 20 May 2018. [Online]. Available: https://github.com/spotify/annoy

[21] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "TensorFlow: A System for Large-Scale Machine Learning," in *OSDI*, vol. 16, 2016, pp. 265–283.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[23] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*, 2017, pp. 1321–1330.

[24] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," *Journal of Machine Learning Research*, vol. 6, no. 4, pp. 503–556, 2005.

[25] H. Van Seijen, H. Van Hasselt, S. Whiteson, and M. Wiering, "A theoretical and empirical analysis of Expected SARSA," in *Adaptive Dynamic Programming and Reinforcement Learning, 2009. ADPRL'09. IEEE Symposium on*. IEEE, 2009, pp. 177–184.

[26] D. Ormoneit and Ś. Sen, "Kernel-based reinforcement learning," *Machine learning*, vol. 49, no. 2-3, pp. 161–178, 2002.

[27] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems*, 1994.

[28] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," *arXiv preprint arXiv:1708.02596*, 2017.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[31] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft, "Learning and policy search in stochastic dynamical systems with Bayesian neural networks," *arXiv preprint arXiv:1605.07127*, 2016.

[32] J. Hernandez-Lobato, Y. Li, M. Rowland, T. Bui, D. Hernandez-Lobato, and R. Turner, "Black-box alpha divergence minimization," in *International Conference on Machine Learning*, 2016, pp. 1511–1520.

[33] D. Maclaurin, D. Duvenaud, and M. Johnson, "Autograd," accessed 21 May 2018. [Online]. Available: https://github.com/HIPS/autograd

[34] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," in *Reinforcement Learning*. Springer, 1992, pp. 5–32.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[36] M. Müllner, B. Urbanek, C. Havel, H. Losert, G. Gamper, and H. Herkner, "Vasopressors for shock," *The Cochrane Library*, 2004.

[37] N. Prasad, L.-F. Cheng, C. Chivers, M. Draugelis, and B. E. Engelhardt, "A reinforcement learning approach to weaning of mechanical ventilation in intensive care units," *arXiv preprint arXiv:1704.06300*, 2017.

[38] S. Nemati, M. M. Ghassemi, and G. D. Clifford, "Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach," in *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, August 2016.

Appendix A
Physiological features in state-space

The physiological features defining the state space are presented here.

Demographics/Static: Shock Index, Elixhauser, SIRS (Systemic Inflammatory Response Syndrome), Gender, Re-admission, GCS (Glasgow Coma Scale), SOFA (Sequential Organ Failure Assessment), Age

Lab Values: Albumin, Arterial pH, Calcium, Glucose, Hemoglobin, Magnesium, PTT (Partial Thromboplastin Time), Potassium, SGPT (Serum Glutamic-Pyruvic Transaminase), Arterial Blood Gas, BUN (Blood Urea Nitrogen), Chloride, Bicarbonate, INR (International Normalized Ratio), Sodium, Arterial Lactate, CO_2, Creatinine, Ionised Calcium, PT (Prothrombin Time), Platelets Count, SGOT (Serum Glutamic-Oxaloacetic Transaminase), Total bilirubin, White Blood Cell Count

Vital Signs: Diastolic Blood Pressure, Systolic Blood Pressure, Mean Blood Pressure, $PaCO_2$, PaO_2, FiO_2, PaO_2/FiO_2 ratio, Respiratory Rate, Temperature (Celsius), Weight (kg), Heart Rate, SpO_2

Intake and Output Events: Fluid Output (4 hourly period), Total Fluid Output, Mechanical Ventilation

Appendix B
Risk Assessment

This project has been entirely computational; as a result, the principal hazard identified before commencing was computer use. I believe this was, overall, an accurate assessment, as there was no laboratory work conducted over the course of the project.

However, with regards to the computer work involved in this project, I think the risk assessment could have been more thorough. As I had to spend a significant amount of time programming for this project, I found that at the end of longer work sessions, my shoulders and back were slightly painful. This most likely arose from poor posture and improper seat positioning at the desk in my room, where I conducted the majority of my work. In addition, as the majority of work was completed using my laptop, I used the inbuilt keyboard and mouse, which may have contributed to the problem – I was not able to assume a more relaxed and natural position while working.

I did tackle these ergonomic issues two months after starting the project – I began to use the department-provided keyboard, mouse, and laptop stand while working, which improved my posture and prevented back/shoulder pain. This was a valuable learning experience for me, and I will bear it in mind in the future when working on computationally-intensive projects.

Although I managed to solve these ergonomic issues, I feel that the original risk assessment could have detailed these problems in some more depth, which would have prevented any back/shoulder discomfort in the first place. A proper assessment of the ergonomic risks involved with extended computer use is therefore what I would have done differently were I to conduct a risk assessment again. In particular, I could have researched more about the best desk setup for a computationally-intensive laptop-based project, which would have led to me to find out that an external keyboard and mouse, and a laptop stand, are effective in reducing ergonomic strain.